Cinematic Behavior Transfer via NeRF-based Differentiable Filming

Xuekun Jiang[1*] Anyi Rao[2*] Jingbo Wang[1] Dahua Lin[1,3] Bo Dai[1]
[1]Shanghai AI Lab [2]Stanford University [3]Chinese University of Hong Kong
[*] denotes equal contribution
anyirao@stanford.edu {jiangxuekun, wangjingbo, daibo}@pjlab.org.cn

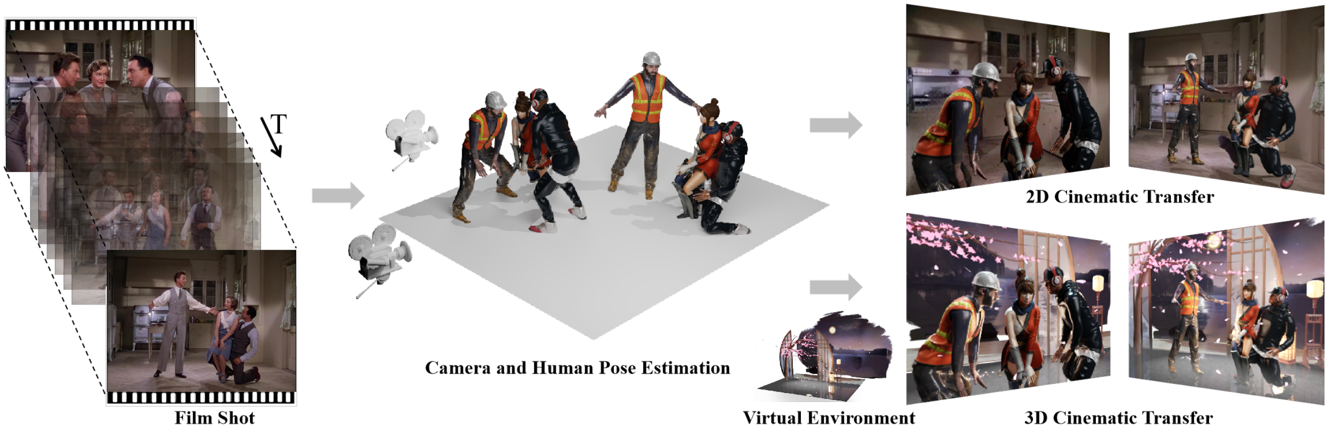


Figure 1. Given a film shot, a series of visual continuous frames, containing complex camera movement and character motion, we present an approach that estimates the camera trajectory and character motion in world coordinates. The extracted camera and characters' behavior can be applied to new 2D/3D content through our cinematic transfer pipeline. 2D cinematic transfer aims to substitute the characters in the original shot with new 3D characters while preserving the identical character motion and camera movements. 3D cinematic transfer can apply the movements of characters and cameras to new characters and scenes, providing more flexibility to modify various properties, such as lighting, character motion, and camera trajectory.

Abstract

In the evolving landscape of digital media and video production, the precise manipulation and reproduction of visual elements like camera movements and character actions are highly desired. Existing SLAM methods face limitations in dynamic scenes and human pose estimation often focuses on 2D projections, neglecting 3D statuses. To address these issues, we first introduce a reverse filming behavior estimation technique. It optimizes camera trajectories by leveraging NeRF as a differentiable renderer and refining SMPL tracks. We then introduce a cinematic transfer pipeline that is able to transfer various shot types to a new 2D video or a 3D virtual environment. The incorporation of 3D engine workflow enables superior rendering and control abilities, which also achieves a higher rating in the user study.

1. Introduction

With the ongoing increase in media consumption, creators are consistently exploring innovative techniques to enhance the viewing experience, reduce production costs, and create compelling narratives. Thus, the ability to manipulate and reproduce specific visual elements, such as camera movements and character behaviors, has long been a sought-after capability in the realm of digital media and video production, as it helps maintain continuity and transfer a particular style or a unique mood from one scene to another.

It is challenging to replicate specific camera and character behaviors across different video clips. Manually achieving it can be both time-consuming and prone to inconsistencies. Alternatively, one can adopt SLAM [4, 23, 24] and SMPL estimation [13] to respectively recover camera poses and human poses, yet they struggle to handle complex scenarios with both camera and character behaviors. Although SLAM and SMPL estimation can be used together [7, 18, 31] to infer both camera and character behaviors, it still has problems of mismatch with the original shot due to the noise caused by dynamic content. Recently, some researchers [28, 32] took advantage of NeRF [15] to inverse-optimize camera poses with fewer effects from dynamic content. Nevertheless, they required manual preparation of a similar scene as NeRF training data, which significantly diminishes their flexibility and scalability.

To address the above challenges, we propose a filming behavior transfer pipeline, that estimates SMPL tracks and camera trajectory of a film shot. The SMPL track is a sequence of human pose. We detect the character motion in reference shot and approximate it by predicting SMPL tracks. While SMPL tracks can be estimated as in previous methods [31], we train a dynamic NeRF [19] to represent the 3D SMPL tracks. We then take NeRF as a differentiable renderer to provide image-level matching supervision for camera trajectory optimization. The optimized camera trajectory can further refine the SMPL tracks, leading to a more accurate estimation of character and camera behaviors. As the above SMPL tracks do not have textures, to meet the artist's workflow needs, we further develop a 3D engine-based workflow to adapt the SMPL track and camera to new virtual characters, enabling a higher level of control and precision in the creative process, such as changing the lighting or adjusting the speed of the camera movement. With this, we can transfer a variety of shot types, including different shot scales, angles, complex camera movements, and various character numbers, which helps artists create new content with similar cinematic behavior.

Extensive experiments show the capacity of our method to extract reasonable character motions and camera trajectory from a given well-known movie shot and generate new content with a similar cinematic style through a 3D engine workflow.

2. Related Work

Human and Camera Motion Estimation. Extracting human and camera motion from video has attracted increasing attention from researchers in recent years. Most recent methods [3, 6] were just focused on how to estimate the human pose in 3D because of the fixed camera. For dynamic camera trajectory, some approaches [9, 34, 35] have tried to circumvent the issue of camera motion by recovering the human trajectories in global coordinates from the per-frame local human poses. Other researchers [5, 7, 12, 18, 31] have introduced SLAM system into human pose estimation to reconstruct the 4D human pose. Pavlakos *et al.* [18] proposed a method to reconstructed 3D humans and environments in TV shows. They used COLMAP and NeRF to reconstruct the cameras and dense scene and use this information to recover accurate 3D pose and location of people over shot boundaries and on monocular frames. Ye *et al.* [31] proposed a method to reconstruct global human trajectories from videos in the wild. They showed that relative camera estimates along with data-driven human motion priors can resolve the scene scale ambiguity and recover global human trajectories. Kocabas *et al.* [7] proposed to tightly integrate SLAM and human motion priors in an optimization that is inspired by bundle adjustment. Unlike the above methods that used SLAM as initialization, our method optimizes camera trajectories by leveraging NeRF as a differentiable renderer.

NeRF-based Camera Pose Estimation. NeRF [16] is a popular representation of 3D scenes, which uses a multilayer perceptron (MLP) that evaluates a 5D implicit function estimating the density and radiance emanating from any position in any direction. Yen *et al.* [32] first proposed to estimate mesh-free camera pose by "inverting" a NeRF. A lot of recent work [1, 2, 8, 10, 14, 25] focused on how to get camera parameters without using SFM, and instead train both camera parameters and NeRF during training using only pictures. Most of them cared more about the quality of the NeRF than the quality of the camera. And iNeRF [32] is limited to its slow inference speed and it's very sensitive to the initial parameters of the camera. To address these problems, Lin *et al.* [11] improved it via 1) using Instant NGP to replace the native NeRF [17]; 2) introducing parallel Monte Carlo sampling to overcome local minima and improved efficiency in a more extensive search space of camera parameters. Wang *et al.* [33] proposed a feature-driven cinematic motion transfer technique. It replicated the camera sequences from movies to a trained NeRF to let the generated video clip maximize the similarity with the reference clip through a designed cinematic loss. Most recent works required manual preparation of a similar scene as NeRF training data. Our approach predict the SMPL tracks from original shot and use a dynamic NeRF to represent it. Hence, our approach eliminates the need for manual scene construction.

3. Method

In this paper, we propose a method to transfer character and camera behavior from a given single shot to new 2D/3D content. For each shot, we extract its SMPL tracks representing the sequence of characters' motion in world coordinates, and then optimize the camera trajectory based on the SMPL tracks, as detailed in Sec. 3.1 and Sec. 3.2. Finally, we used the SMPL tracks and the camera trajectory to create new content through a full 3D engine workflow in Sec. 3.3.

3.1. Human Pose and Camera Estimation

Due to the coupling of character and camera movement in a video, it is hard to obtain accurate human motions in world coordinates. We need to estimate a camera trajectory in the world coordinates to decouple the human and camera motions to eliminate the ambiguity of scene scale in camera space. Typically, we utilize SLAM (simultaneous localization and mapping) to extract the camera trajectory. Given a single shot V with T frames, $V = \{I_1, \ldots, I_T\}$ with N characters, we first predict a starting camera trajectory $\hat{C} = \{\hat{c}_t\}_{t=1}^{T}$ in world coordinates with SLAM method. And then we predict N SMPL tracks $S_c = \{S_{c,n}\}_{n=1}^{N}$ in camera

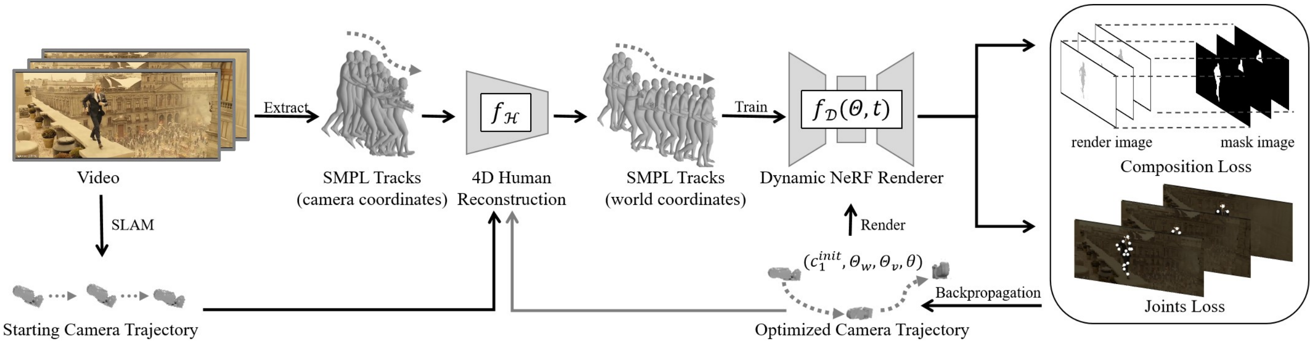


Figure 2. The overview of our differentiable pipeline of characters' motion and camera trajectory estimation. Given a shot video, we first extract SMPL tracks in camera coordinates and a camera trajectory in world coordinates. Then, we reconstruct the motions of all characters in world coordinates by a 4D human reconstruction method. Finally, we optimize camera trajectories by leveraging NeRF as a differentiable renderer and refine SMPL tracks.

coordinates. N means the number of people in video. With S_c and $\hat{C}$, we can compute the SMPL tracks S_w in world coordinates with a 4D human reconstruction method $f_{\mathcal{H}}$,

$$S_w = f_{\mathcal{H}}(S_c, \hat{C}). \tag{1}$$

Although the starting camera trajectory $\hat{C}$ predicted from V can resolve the scene scale ambiguity and help us to recover the human motions in world coordinates, $\hat{C}$ suffers from the errors caused by the dynamic content in V. We propose to optimize a new camera trajectory $C^* = \{c_t^*\}_{t=1}^{T}$ based on S_w via a differential render NeRF to add image-level supervision.

NeRF represents a 3D scene in a differentiable way that can render an image with a given camera pose. This also means that, for a trained NeRF, we can figure out an optimal camera pose that renders frames that match our reference shot V. Towards this goal, we train a dynamic NeRF $f_{\mathcal{D}}(\Theta, t)$ to capture the character motion tracks S_w. Hence, the camera trajectory optimization can be treated as an inverted optimization, which takes a trained NeRF as a differentiable render and uses the NeRF backpass gradients to find the optimal camera parameters.

$$c_t^* = \underset{c_t \in \mathrm{SE}(3)}{\mathrm{argmin}} \mathcal{L}(c_t \mid I_t, \Theta), \tag{2}$$

where Θ is the parameters of NeRF, I_t is the reference shot frame image in time t. Due to the lack of background and details of the approximation scene of SMPL, our method can not compute the loss directly from the RGB images like iNeRF [32], and optical flow or human pose like JAWS [28]. To tackle the above issues, we introduce two losses: a composition loss $\mathcal{L}_c$ and a joint loss $\mathcal{L}_j$. i) The composition loss $\mathcal{L}_c$ is calculated by an instant mask image for the original image and the NeRF rendered image. For the mask images, We color the pixels of each human mask to match the vertex color of the corresponding SMPL model and color the rest of pixels white. Due to the lack of detail in SMPL, the clothes of the people will affect the results, especially some special clothing styles such as ornate gowns. To solve this problem, we introduce the second loss. ii) The joint loss $\mathcal{L}_j$ is calculated by the joint distances between the original image and the rendered image. For the original image, we predict the joints 2D coordinates of each character as the ground truth by ViTPose [30]. For the rendered image, we reproject the SMPL joints in 3D to 2D by optimized camera pose. The final loss is a weighted sum of $\mathcal{L}_c$ and $\mathcal{L}_j$.

After the optimization, we obtain a more accurate camera trajectory C^*. Compared to the starting camera trajectory $\hat{C}$ predicted solely by SMPL, it can better align the reconstructed image framing with the original shot by using NeRF as the differentiable renderer with two image-level losses. Since a more reasonable camera trajectory leads to better SMPL tracks in world coordinates, we can update Eq. (1) and compute more accurate SMPL tracks,

$$S_w^* = f_{\mathcal{H}}(S_c, C^*). \tag{3}$$

3.2. Camera Trajectory Optimization

A key step in the above optimization is to optimize the camera trajectory. Nevertheless, using NeRF to directly regress the values in the transformation matrix c^* does not guarantee that the optimized result remains in the $SE(3)$ manifold.

Preliminary on optimiazation parameters. In most NeRF-based pose estimation works [32], the camera pose is defined as a transform matrix $c^* \in SE(3)$ in world coordinates, and is estimated by a trained NeRF model. To ensure the estimated pose still lies in the $SE(3)$ manifold during gradient-based optimization, the camera pose c^* is represented by an initial pose estimate $c^{\text{init}} \in SE(3)$ and a transformation matrix A with exponential coordinates:

$$\begin{aligned} & c^* = A c^{\text{init}}, \\ & \text{where } A = e^{[S]\theta} = \begin{bmatrix} e^{[w]\theta} & f_{\mathcal{K}(\theta,w,v)} \\ 0 & 1 \end{bmatrix}, \\ & f_{\mathcal{K}(\theta,w,v)} = (I\theta + (1-\cos\theta)[w] + (\theta - \sin\theta[w]^2))v. \end{aligned} \tag{4}$$

Here $S = [w, v]^T$ represents the screw axis, θ is a magnitude, $[w]$ represents the skew-symmetric 3 × 3 matrix of w. The matrix optimization problem is then equivalent to figuring out the optimal parameters (θ, w, v). With this parameterization according to Eq. (4), the optimal goal is to achieve optimal relative transformation from an initial estimated pose c^{init}:

$$\theta, w, v = \underset{S\theta \in \mathbb{R}^6}{\operatorname{argmin}} \, \mathcal{L}(Ac^{\text{init}} \mid I, \Theta). \tag{5}$$

For each given observed image, the camera parameters (θ, w, v) are initialized near 0, and each is drawn from random from a zero-mean normal distribution $\mathcal{N}(0, \sigma = 10^{-6})$.

Sequence camera parameters optimization. Existing NeRF-based camera pose estimation works [28, 32] focus on single-camera pose estimation and rarely tackle a sequence. Their primary objective is to ensure that the rendered image resulting from camera optimization closely resembles the target image. Consequently, they often prioritize this visual similarity over the accuracy of the trajectory in 3D world coordinates. They optimized each time camera parameter c_t^* independently during this process.

We aim to optimize the camera pose to produce a correct 2D image and predict the correct inverted 3D camera trajectory. Regrettably, constructing a 3D scene and obtaining a NeRF representation identical to that of a movie shot with complex dynamic content significant challenges. We accurately detect the pose of the character in the reference shot and subsequently estimate it by predicting SMPL tracks. However, the introduction of noise through SMPL predictions amplifies errors in camera pose estimation, leading to an unreasonable camera trajectory in 3D world coordinates. These motivate us to learn a continuous representation of the camera trajectory to prevent mutations.

As mentioned in Eq. (4), the camera trajectory parameters $c_t^* \in C^*$ for each time step t can be decomposed by an initial pose c_t^{init} and a transformation matrix A_t

$$c_t^* = A_t c_t^{\text{init}}. \tag{6}$$

To learn a continuous representation of the camera trajectory, we use two strategies. First, the initial pose c_t^{init} is derived from the camera parameters of the preceding moment: $c_t^{\text{init}} = c_{t-1}^*$. Second, we defined a continuous function $f_{\mathcal{A}}$ with respect to time t to calculate transformation matrix A_t: $A_t = f_{\mathcal{A}}(t)$. The matrix A_t is defined by parameters (θ, w_t, v_t) according to Eq. (4). The parameter θ remains constant throughout the entire camera trajectory Therefore, as long as the parameters w_t and v_t are continuous, the continuity of the camera trajectory can be guaranteed. We use two MLP $f_{\mathcal{W}}$ and $f_{\mathcal{V}}$ to predicted the w_t and v_t in each time step t:

$$\begin{aligned} w_t &= w_1 + f_{\mathcal{W}}(t), \\ v_t &= v_1 + f_{\mathcal{V}}(t). \end{aligned} \tag{7}$$

As mentioned in Sec. 3.1, the S_w is projected to the world coordinates by SLAM camera $\hat{C}$, and then we used the S_w to optimize the camera trajectory C^*. The camera trajectory C^* is aligned to the SLAM camera $\hat{C}$. To calculate the parameters θ, w_1 and v_1 in the first time step. So we can use the $\hat{c}_1$ as the c_1^{init} to optimise the first camera pose c_1^* according to Eq. (4) that can reduce the computational cost. We use Eq. (5) to optimize the θ, w_1 and v_1.

Finally, we transform Eq. (6) into the following:

$$c_t^* = f_{\mathcal{A}}(\Theta_w, \Theta_v, \theta, t) c_{t-1}^*, \quad t \in [2, t), \tag{8}$$

where Θ_w and Θ_v is the parameters of MLP $f_{\mathcal{W}}$ and $f_{\mathcal{V}}$.

3.3. Transfer via a 3D Engine-based Workflow

With the accurate character and camera behavior estimation, we can transfer them to 2D and 3D content which are shown in Fig. 3 (d) and (e).

2D cinematic transfer aims to replace characters in an existing film shot with new 3D characters, while keeping the same character motions and camera behavior. For 2D cinematic transfer, we first render a pure video without background V_f with our camera trajectory and the character after retargeting. Then, we remove the foreground characters from the original shot. In this paper, we use an advanced object removal method ProPainter [36], to erase the characters and generate a pure background video V_b. We combine these two videos V_f and V_b to obtain the final results.

3D cinematic transfer takes character and camera movements and applies them in new characters and scenes, which further allows for adjustments in different aspects like lighting, character motion, camera motion, offering more control and options for personalizing the final result. 3D cinematic transfer is much simpler, which only needs to apply the motion and camera to the virtual scene and render the result. However, compared to the 2D workflow, 3D cinematic transfer has more flexibility. Since the entire scene is in the 3D space, we are free to modify it according to our needs, for example, change the time from night to day (Fig. 3 middle (e)), place a robot in the corner (Fig. 3 below (e)).

4. Experiments

4.1. Implementation Details.

Our implementation is based on ‘torch-ngp’ and Pytorch. We use PHALP [20] for human pose tracking, SLAHMR [31] for 4D human reconstruction, D-NeRF [19] for neural rendering and VitPose [30] for 2D joints prediction. More details are shown in the supplementary.

4.2. Qualitative Results

Our approach can reproduce character movements and camera trajectories from a given shot, replace characters in the

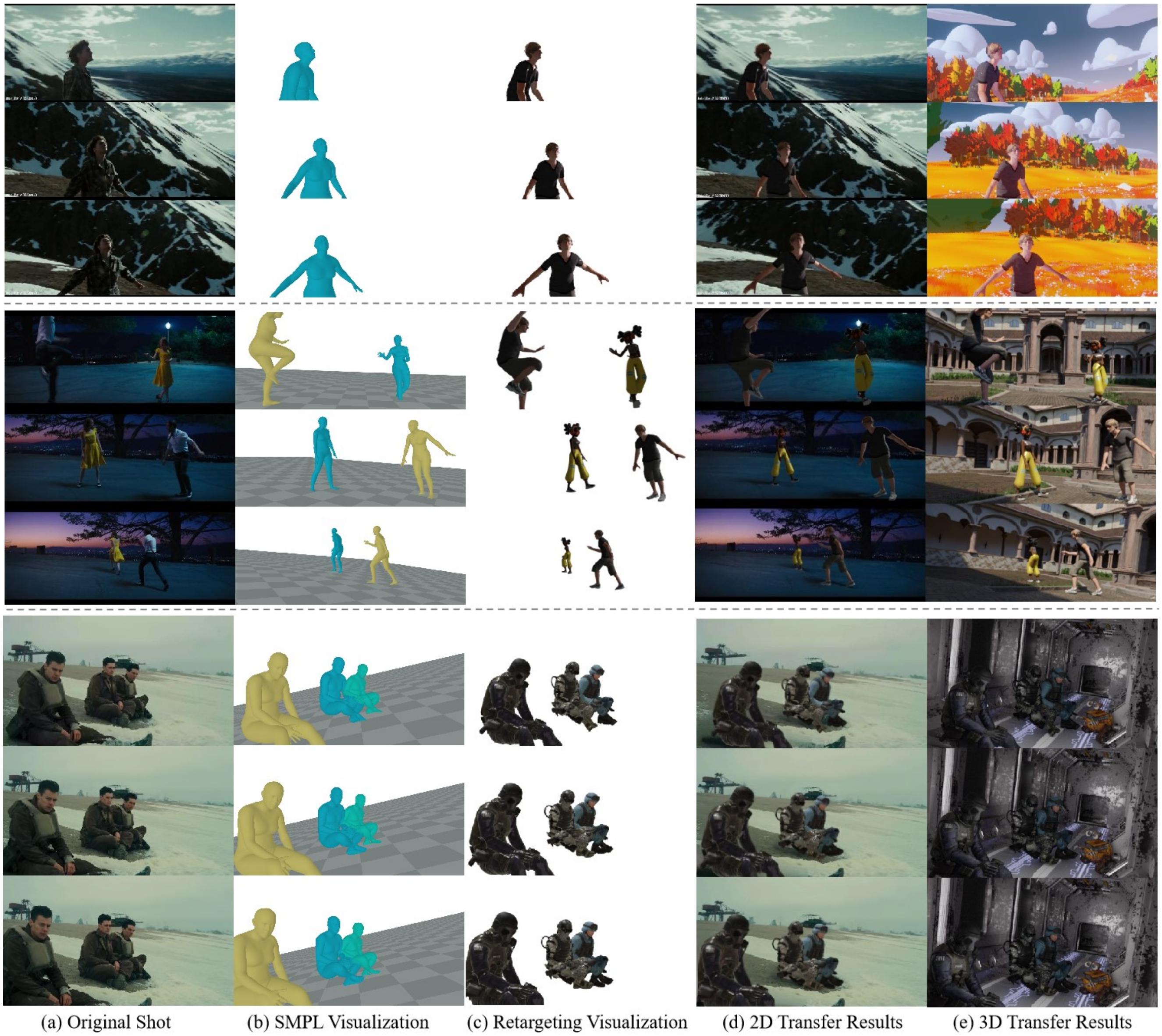
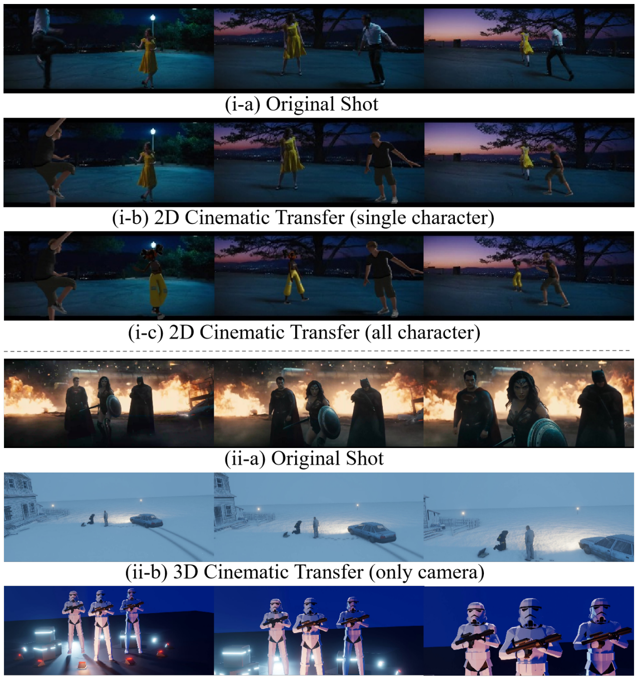


Figure 3. Examples of cinematic behavior transfer. (a) The original shot: we present three common shot types Arc, Track, and Push In. (b) SMPL visualization of our method. We recreate the cinematic behavior by extracting the SMPL tracks and the camera trajectory from the original shot. (c) New Characters retargeting visualization of our method. We apply the motion of SMPL to new character and render images with optimized camera trajectory through our engine workflow. (d) The 2D cinematic transfer results. We erase the characters in the foreground and combine the background video with (c) to generate a new 2D video like (a). (e) The 3D cinematic transfer results. We apply the motion and camera to a new virtual scene, like a cartoon grassland, palace, and SF tunnel.

frame, or change the scene through an engine-based workflow. Fig. 3 showcases some examples of cinematic transfer from an original film shot to a synthetic environment. Compared with other methods, our method can handle various scenes with single/double/multiple characters. It can also recover various types [21, 22] of shot trajectories, such as Arc, Track, and Push-In.

Specifically, in Fig. 3: (a) The original shot: we present three common shot types Arc, Track, and Push-In. Arc shot orbits the camera around a subject in an arc pattern. Track shot moves the camera through the scene for an extended amount of time. Push-In shot moves the camera closer to a subject. (b) SMPL visualization of our estimation. We extract and optimize the SMPL tracks and the camera trajectory from the original shot, which can reproduce the characters' motion and the camera movement. (c) New 3D characters' retargeting visualization. We retarget the original shot's SMPL motion to new 3D characters and render images with optimized camera trajectory through our engine workflow. This allows us to further put them into a scene. (d) The 2d cine-


(i-a) Original Shot

(i-b) 2D Cinematic Transfer (single character)

(i-c) 2D Cinematic Transfer (all character)

(ii-a) Original Shot

(ii-b) 3D Cinematic Transfer (only camera)

(ii-c) 3D Cinematic Transfer (char. & camera)

Figure 4. Flexibility enabled by 3D engine workflows. For 2D cinematic transfer, we can replace any character like (i-b) or (i-c), in shot (i-a). For 3D cinematic transfer, we demonstrate the ability to apply either the camera alone (ii-b) or both character and camera information to new content(ii-c).

matic transfer results. We replace any character in the original shot. For instance, we can replace "Mia" and "Sebastian", the characters in *Lalaland*, with our own 3D characters or keep "Mia" and replace "Sebastian" with a new 3D character as shown in Fig. 4 (i-b). It is implemented by first using an advanced object removal method [36] to erase the character in the foreground and combining the background video with (c) to generate the final video. (e) The 3d cinematic transfer results. We apply the characters' motion and camera movement to new 3D characters and a new virtual scene. This provides the user with more freedom of operation, *e.g.*, adjusting the lighting, or modifying the camera.

Flexibility of our cinematic transfer. Fig. 4 shows various transfer results with our 3D engine-based workflow. After investigating artist's workflows, we figured out that to truly meet their needs, the workflow should enable freely using different information extracted from the video, such as the motions of the characters or the movement of the camera. Our approach can well provide this flexibility to explore various creative possibilities: replacing one character and keeping the others within a movie shot (Fig. 4 (i-c)); employing solely the camera trajectory from a movie shot to a new scene with different character motions (Fig. 4 (ii-b)). However, the recent SOTA cinematic transfer method JAWS [28] does not fully support this flexibility. Their workflow can only apply camera trajectory to scenes that closely resemble the reference video in terms of the number of characters and their relative positions.

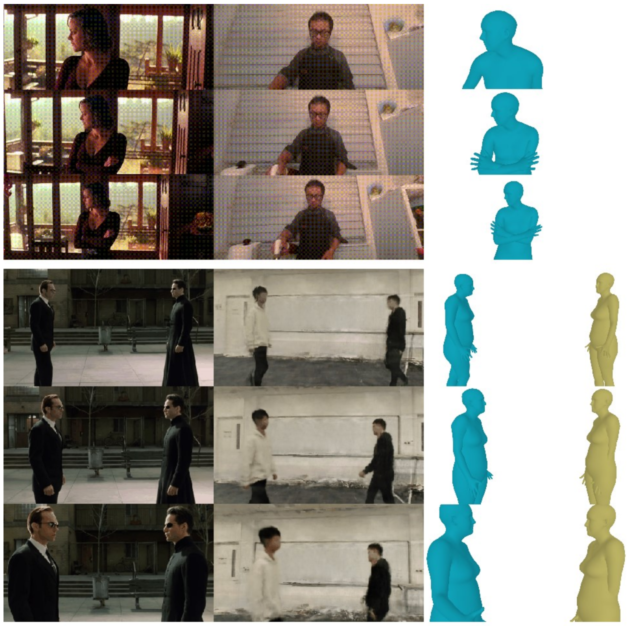

(a) Original Shot (b) JAWS (c) Ours

Figure 5. Comparison of the cinematic transfer results with SOTA. Our method demonstrates a better ability to align with the composition of the original shot compared to JAWS [28]. The two shots are from *Inception, 2010* (top) and the *Matrix, 1999* (below)

Comparison with SOTA in cinematic transfer. JAWS [28] is an optimization-driven approach that addresses the cinematic transfer from a reference clip to a trained NeRF. It used a *on-screen* loss and a *iner-frame* loss to cover the framing and camera motion aspects. It is limited to handling highly mismatched character poses due to inter-frame motion. So, it has to hand-craft a scene that is essentially the same as the original shot to train NeRF, which greatly limits its usage scenarios. Additionally, simply adapting JAWS to our setting does not work due to the lack of background and details of the predicting SMPL tracks. To be specific, the RAFT [26] they used for the optical flow estimation cannot work for the rendered SMPL image and LitePose [29] they used to infer the post joint will ignore the inter-frame motion. To achieve a strong comparison with JAWS, we use the shots used in their papers' experiments, as shown in Fig. 5. Our method restores not only the composition of the shot but also the action of the characters. Although JAWS used a realistic environment similar to the original shot, it does not accurately reproduce the composition of the original shot. We can clearly observe the flaws in the "Matrix" example, where two characters occupy the left and right parts of the image when the camera is pushed to the end. The composition of the characters in JAWS's results do not closely align with the original footage. Another limitation of JAWS is that it highly relies on dynamic NeRF results, which will easily fail on the complex motion shots that dynamic NeRF cannot

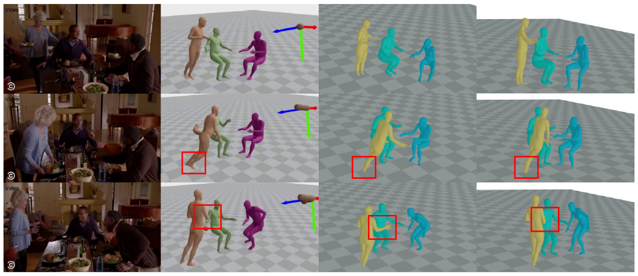


Figure 6. Comparison of the human pose estimation in world coordinates with SOTA. As shown in the red box, our method has better results in limb details due to the improved optimization of the camera trajectory.

handle. As our method does not require background details of the predicting SMPL tracks, our approach has lower quality requirements for NeRF's quality, which leads to stronger robustness and better performance.

Comparison with SOTA in world coordinates human pose estimation. As we mentioned in Sec. 3.1, when we finished the the camera C^* optimization with $f_{\mathcal{D}}$, we can bring it into $f_{\mathcal{H}}$ and get new SMPL tracks. Fig. 6 compares the human pose estimation results with SOTA methods. Due to the enhanced optimization of the camera trajectory, our method achieves better results in capturing detailed limb poses. Inspired by bundle adjustment, PACE [7] tightly integrates SLAM and human motion priors in optimization. It can handle the shot with that entire character's body but is limited to dealing with the shot where only the character's partial body appears. To achieve a strong comparison with PACE, we use the shots used in their papers' experiments. In Fig. 6 (b), the feet of the figure on the left are not on the ground. Since films contain lots of partial body shots like close-up shot or medium shot, PACE is not very suitable for cinematic transfer. SLAHMR [31] used relative camera estimates along with data-driven human motion priors to resolve the scene scale ambiguity and recover the human trajectories in world coordinates. However, the human pose is likely to fail due to the noise from DROID-SLAM on dynamic content. In the leftmost character shown in the red box of Fig. 6 (c), the arms should be close to the body, while SLAHMR predicts open arms. We do not directly use SLAM's camera trajectory and utilize NeRF as a differential render to re-optimize the camera pose. Due to the improved optimization of the camera trajectory, our method achieves better results in capturing finer details of limb poses.

4.3. Quantitative Results

Since A key feature in cinematic behavior transfer is to check if the transferred results are similar to the original shot, we test three metrics on the frame composition restoration and conduct a user study to validate users' satisfaction with 2D and 3D transfer.

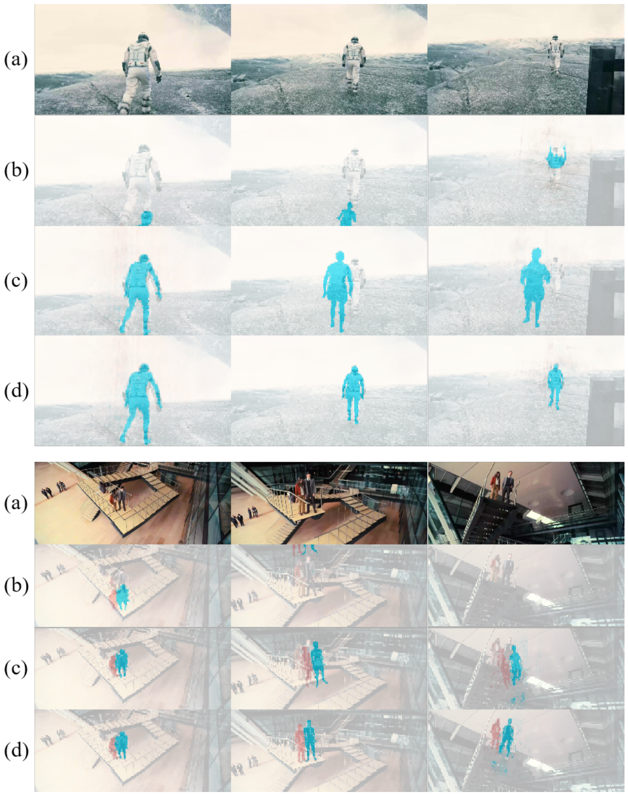


Figure 7. We show the results of the alignment visualization: (a) the original shot, (b) DROID-SLAM, (c) iNeRF, (d) our method.

Comparison with SOTA on frame composition restoration. Tab. 1 shows the quantitative results between our method with two SOTA camera pose estimation methods DROID-SLAM [27] and iNeRF [32]. DROID-SLAM is a deep learning-based SLAM system with fewer catastrophic failures. iNeRF uses NeRF for mesh-free, RGB-only 6DoF camera pose estimation, which uses RGB pixels as a supervision signal. Due to the lack of background and details of SMPL, we use the composition loss $\mathcal{L}_c$ instead of RGB loss to implement iNeRF. We test more than 100 well-known film shots that collected from the Internet with multiple styles to show the effectiveness of our method.

To measure the effect of different methods on restoring picture composition, we use three metrics to evaluate all the methods: 1) Pixel Accuracy (PA): It is the percentage of pixels in the segmentation image that are correctly classified. 2) Intersection over Union (IoU): The overlap between the character segmentation map in the rendered shot and the character segmentation map in the original shot. 3) Mean Per Joint Position Error (MPJPE): The mean Euclidean distance between the predicted key bone point and the true value. As shown in Tab. 1, (i) Changing backgrounds and moving figures can greatly affect SLAM's accuracy. It can be seen from the three metrics that the position of the characters in the picture rendered by SLAM is greatly offset from the

Shot Move.	PUSH-IN			PULL-OUT			PAN		
Methods	PA↑	IoU↑	MPJPE↓	PA↑	IoU↑	MPJPE↓	PA↑	IoU↑	MPJPE↓
DROID-SLAM [27]	86.2	85.8	404.9	86.0	85.3	356.2	91.9	89.6	40.9
iNeRF [32]	89.0	88.2	292.6	92.8	91.5	83.9	83.9	81.0	109.6
Ours	**89.9**	**88.5**	**59.6**	**94.8**	**94.0**	**23.8**	**93.4**	**91.4**	**21.4**
Shot Move.	TRACK			FOLLOW			ARC		
Methods	PA↑	IoU↑	MPJPE↓	PA↑	IoU ↑	MPJPE↓	PA↑	IoU↑	MPJPE ↓
DROID-SLAM [27]	89.3	88.3	109.2	73.3	70.5	1046.9	92.7	92.6	145.2
iNeRF [32]	90.1	89.2	58.5	85.3	85.1	267.5	90.8	90.5	116.3
Ours	**94.5**	**93.8**	**21.8**	**91.3**	**90.5**	**130.9**	**94.8**	**94.5**	**47.9**

Table 1. Comparison with the state-of-the-art camera pose estimation methods on different shot movement types. Our approach outperforms the other baselines across all metrics.

Methods		SLAM+SMPL	Ours
2D restoration	camera mov.	4.7±1.3	6.0±0.5
	char. motion	5.5±1.1	5.8±0.8
3D restoration	camera mov.	4.4±1.0	5.3±0.6
	char. motion	4.9±0.9	5.0±1.0

Table 2. Pair-wise comparison of our method and baseline on content restoration in seven-point Likert scale (lowest-highest:1-7).

original shot, which means that the SLAM method cannot get the correct camera trajectory. (ii) Since iNeRF only uses RGB pixels as loss, it can not restore the composition of the picture very well. (iii) Our method achieves the best results in all metrics, which shows that our method can accurately restore the composition of the original shot, and the extracted camera trajectory is basically correct.

Fig. 7 visualizes the rendered image of different methods. It can be clearly seen that both SLAM and iNeRF rendered images in which character's positions are obviously offset from the original images. The images rendered by our method are very consistent, indicating that the camera trajectory we obtained is correct.

User study. To further demonstrate the validity of our method in practice, we conduct a user study on 30 shots from different films among 10 volunteers. Our study focuses on the accurate recovery of the original video on the screen, and the results of the restored characters and camera movements in world coordinates. Volunteers are required to compare the original shot with the 2D and 3D results and to determine how well the two matched up with the seven-point Likert scale (lowest-highest:1-7). To have a strong baseline, we combined DROID-SLAM [27] and SMPL [31] estimation to jointly infer camera and character behaviors. Volunteers were asked to view the original shot and the results of both methods at the same time, and to rate both results. For 2D results, we use the rendered result from camera view like Fig. 5 (c). For 3D, we chose a side view that can see the movement of the character and the camera completely like Fig. 6 (d). In order to make it easier for volunteers to make judgments, we will provide more than two side views.

Tab. 2 shows that: (i) By employing the NeRF technique to re-optimize the camera trajectory, our method received positive feedback from users who perceiving the extracted camera trajectory as more reasonable compared to SLAM method. (ii) Due to the more reasonable camera trajectory to refine SMPL tracks, users have observed enhanced poses in our optimized SMPL, which in turn has created a greater sense of consistency with the original shot characteristics. It is important to acknowledge that in movie shots, the character's body is often partially visible, which may lead to an accurate visual representation but lacks accuracy in 3D space. For example, the feet may not be properly positioned on the ground but suspended in the air.

5. Discussion and Conclusion

In comparison to previous works, our method exhibits improved performance and robustness across a wide range of scenes. However, our approach still has certain limitations: i) Our approach relies on a starting camera trajectories obtained from SLAM technology to acquire SMPL tracks. Consequently, when the content of a shot changes too rapidly to extract SMPL tracks, our method is unable to produce the correct results. ii) Our approach is specifically designed for shots that prominently feature human subjects. However, in scenarios where the primary focus of a shot shifts towards showcasing the environment or objects, our method transitions into a simplified version resembling a SLAM approach.

We introduce a reverse filming behavior estimation technique that enables cinematic behavior transfer. It utilizes NeRF as a differentiable renderer, effectively optimizes camera trajectories and refines character movements with SMPL models. Additionally, our innovative cinematic transfer pipeline demonstrates its versatility by efficiently transferring various shot types to both 2D video and 3D virtual environments. The integration of a 3D engine workflow not only enhances rendering quality and control but also garners a higher user satisfaction rating, showcasing the potential of our approach in digital media production.

Acknowledgement. This work is funded in part by National Key R&D Program of China (2022ZD0160201), and Shanghai Artificial Intelligence Laboratory.

References

[1] Zezhou Cheng, Carlos Esteves, Varun Jampani, Abhishek Kar, Subhransu Maji, and Ameesh Makadia. Lu-nerf: Scene and pose estimation by synchronizing local unposed nerfs. *arXiv preprint arXiv:2306.05410*, 2023. 2

[2] Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Garf: gaussian activated radiance fields for high fidelity reconstruction and pose estimation. *arXiv e-prints*, pages arXiv–2204, 2022. 2

[3] Hongsuk Choi, Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee. Beyond static features for temporally consistent 3d human pose and shape from a video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1964–1973, 2021. 2

[4] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping. *IEEE robotics & automation magazine*, 13 (2):99–110, 2006. 1

[5] Dorian F Henning, Tristan Laidlow, and Stefan Leutenegger. Bodyslam: joint camera localisation, mapping, and human motion tracking. In *European Conference on Computer Vision*, pages 656–673. Springer, 2022. 2

[6] Angjoo Kanazawa, Jason Y Zhang, Panna Felsen, and Jitendra Malik. Learning 3d human dynamics from video. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5614–5623, 2019. 2

[7] Muhammed Kocabas, Ye Yuan, Pavlo Molchanov, Yunrong Guo, Michael J. Black, Otmar Hilliges, Jan Kautz, and Umar Iqbal. Pace: Human and motion estimation from in-the-wild videos. In *3DV*, 2024. 1, 2, 7

[8] Axel Levy, Mark Matthews, Matan Sela, Gordon Wetzstein, and Dmitry Lagun. Melon: Nerf with unposed images using equivalence class estimation. *arXiv preprint arXiv:2303.08096*, 2023. 2

[9] Jiefeng Li, Siyuan Bian, Chao Xu, Gang Liu, Gang Yu, and Cewu Lu. D &d: Learning human dynamics from dynamic camera. In *European Conference on Computer Vision*, pages 479–496. Springer, 2022. 2

[10] Chen-Hsuan Lin, Wei-Chiu Ma, Antonio Torralba, and Simon Lucey. Barf: Bundle-adjusting neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5741–5751, 2021. 2

[11] Yunzhi Lin, Thomas Müller, Jonathan Tremblay, Bowen Wen, Stephen Tyree, Alex Evans, Patricio A Vela, and Stan Birchfield. Parallel inversion of neural radiance fields for robust pose estimation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9377–9384. IEEE, 2023. 2

[12] Miao Liu, Dexin Yang, Yan Zhang, Zhaopeng Cui, James M Rehg, and Siyu Tang. 4d human body capture from egocentric video via 3d scene grounding. In *2021 international conference on 3D vision (3DV)*, pages 930–939. IEEE, 2021. 2

[13] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pages 851–866. 2023. 1

[14] Quan Meng, Anpei Chen, Haimin Luo, Minye Wu, Hao Su, Lan Xu, Xuming He, and Jingyi Yu. Gnerf: Gan-based neural radiance field without posed camera. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6351–6361, 2021. 2

[15] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1

[16] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[17] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2

[18] Georgios Pavlakos, Ethan Weber, Matthew Tancik, and Angjoo Kanazawa. The one where they reconstructed 3d humans and environments in tv shows. In *European Conference on Computer Vision*, pages 732–749. Springer, 2022. 1, 2

[19] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 2, 4

[20] Jathushan Rajasegaran, Georgios Pavlakos, Angjoo Kanazawa, and Jitendra Malik. Tracking people by predicting 3D appearance, location & pose. In *CVPR*, 2022. 4

[21] Anyi Rao, Jiaze Wang, Linning Xu, Xuekun Jiang, Qingqiu Huang, Bolei Zhou, and Dahua Lin. A unified framework for shot type classification based on subject centric lens. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, pages 17–34. Springer, 2020. 5

[22] Anyi Rao, Xuekun Jiang, Yuwei Guo, Linning Xu, Lei Yang, Libiao Jin, Dahua Lin, and Bo Dai. Dynamic storyboard generation in an engine-based virtual environment for video production. In *ACM SIGGRAPH 2023 Posters*, pages 1–2. 2023. 5

[23] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[24] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1

[25] Cameron Smith, Yilun Du, Ayush Tewari, and Vincent Sitzmann. Flowcam: Training generalizable 3d radiance fields without camera poses via pixel-aligned scene flow. *arXiv preprint arXiv:2306.00180*, 2023. 2

[26] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 6

[27] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 7, 8

[28] Xi Wang, Robin Courant, Jinglei Shi, Eric Marchand, and Marc Christie. Jaws: Just a wild shot for cinematic transfer in neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16933–16942, 2023. 1, 3, 4, 6

[29] Yihan Wang, Muyang Li, Han Cai, Wei-Ming Chen, and Song Han. Lite pose: Efficient architecture design for 2d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13126–13136, 2022. 6

[30] Yufei Xu, Jing Zhang, Qiming Zhang, and Dacheng Tao. ViTPose: Simple vision transformer baselines for human pose estimation. In *Advances in Neural Information Processing Systems*, 2022. 3, 4

[31] Vickie Ye, Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Decoupling human and camera motion from videos in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21222–21232, 2023. 1, 2, 4, 7, 8

[32] Lin Yen-Chen, Pete Florence, Jonathan T Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1323–1330. IEEE, 2021. 1, 2, 3, 4, 7, 8

[33] Jung Eun Yoo, Kwanggyoon Seo, Sanghun Park, Jaedong Kim, Dawon Lee, and Junyong Noh. Virtual camera layout generation using a reference video. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pages 1–11, 2021. 2

[34] Ri Yu, Hwangpil Park, and Jehee Lee. Human dynamics from monocular video with dynamic camera movements. *ACM Transactions on Graphics (TOG)*, 40(6):1–14, 2021. 2

[35] Ye Yuan, Umar Iqbal, Pavlo Molchanov, Kris Kitani, and Jan Kautz. Glamr: Global occlusion-aware human mesh recovery with dynamic cameras. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11038–11049, 2022. 2

[36] Shangchen Zhou, Chongyi Li, Kelvin C.K Chan, and Chen Change Loy. ProPainter: Improving propagation and transformer for video inpainting. In *Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2023. 4, 6